The workflow trace archive

Versluis, Laurens; Matha, Roland; Talluri, Sacheendra; Hegeman, Tim; Prodan, Radu; Deelman, Ewa; Iosup, Alexandru

published in
IEEE Transactions on Parallel and Distributed Systems
2020

DOI (link to publisher)
10.1109/TPDS.2020.2984821

document version
Publisher's PDF, also known as Version of record

The Workflow Trace Archive: Open-Access Data From Public and Private Computing Infrastructures

Laurens Versluis, Roland Mathá, Sacheendra Talluri, Tim Hegeman, Radu Prodan, Ewa Deelman, and Alexandru Iosup

Abstract—Realistic, relevant, and reproducible experiments often need input traces collected from real-world environments. In this work, we focus on traces of workflows—common in datacenters, clouds, and HPC infrastructures. We show that the state-of-the-art in using workflow-traces raises important issues: (1) the use of realistic traces is infrequent and (2) the use of realistic, *open-access* traces even more so. Alleviating these issues, we introduce the Workflow Trace Archive (WTA), an open-access archive of workflow traces from diverse computing infrastructures and tooling to parse, validate, and analyze traces. The WTA includes >48 million workflows captured from >10 computing infrastructures, representing a broad diversity of trace domains and characteristics. To emphasize the importance of trace diversity, we characterize the WTA contents and analyze in simulation the impact of trace diversity on experiment results. Our results indicate significant differences in characteristics, properties, and workflow structures between workload sources, domains, and fields.

Index Terms—Workflow, open-source, open-access, traces, characterization, archive, survey, simulation

1 Introduction

Workflows are already a significant part of private datacenter and public cloud infrastructures [1], [2]. This trend is likely to intensify [3], [4], as organizations and companies transition from basic to increasingly more sophisticated cloud-based services. For example, 96 percent of companies responding to RightScale's 2018 survey are using the cloud [5], up from 86 percent in 2012 [6]; the average organization combines services across five *public and private* clouds. To maintain, tune, and develop the *computing infrastructures* for running workflows at the massive scale and with the diversity suggested by these trends, the systems community requires adequate capabilities for testing and experimentation. Although the community is aware that workload traces enable a broad class of realistic, relevant, and reproducible experiments, currently such traces are infrequently used, as we summarize in Fig. 1 (left) and quantify in Section 2. Toward addressing this problem, we focus on improving trace availability and understanding by proposing a new, free and open-access *Workflow Trace Archive (WTA)*, as detailed in Fig. 1 (right) and in the remainder of this work.

The need for workflow traces is stringent [4], [7]. In this work, we adopt the workflow model of Coffman and Graham [8]. In this model, a workflow is considered a directed acyclic graph (DAG) where each vertex represents a task and an edge a computation/data constraint. As such, we do not consider workflow formalisms with iteration (loops) and human interaction, such as BPMN/BPEL [9] and Petri nets [10]. We consider as tasks a broad range of activities, that is, black boxes ranging from simple compute and data operations to entire workflows, recursively.

A workflow trace is a recording of useful, relevant information during the processing of the workflow. Traces can be used to create models with, or used in emulations and simulations to replay the execution of a workflow in a controlled environment, etc. Not only the sheer volume of workloads has increased significantly over time [2], but also the users of datacenters and cloud operations are expecting increasingly better Quality of Service (QoS) from the workflow-management systems, including elasticity, reliability, and low-cost, under strong assumptions of validation [4], [7] and reproducibility [3], [11]. Developing workflow management systems to meet these requirements requires considerable scientific and technical advances and, correspondingly, comprehensive trace-based experimentation and testing. This can be conducted (i) *in vivo*, i.e., experimenting in live/production settings, (ii) *in vitro*, i.e., experimenting using emulation, and (iii) *in silico* i.e., experimenting in simulation [12].

Testing such systems, especially at cluster and datacenter scale, often cannot be done in vivo, due to

- *L. Versluis, T. Hegeman, S. Talluri, and A. Iosup are with Computer Science, Vrije Universiteit Amsterdam, 1081 HV Amsterdam, Netherlands. E-mail: {l.f.d.versluis, A.Iosup}@vu.nl, {sacheendra.t, tim.m.hegeman}@gmail.com.*
- *R. Mathá is with the Institute of Computer Science, Universitat Innsbruck, 6020 Innsbruck, Tyrol, Austria. E-mail: roland@dps.uibk.ac.at.*
- *R. Prodan is with the Institute of Software Technology, University of Klagenfurt, 9020 Klagenfurt am, Austria. E-mail: radu@itec.aau.at.*
- *E. Deelman is with Information Sciences Institute, University of Southern California, Los Angeles, CA 90292. E-mail: deelman@isi.edu.*

Manuscript received 8 July 2019; revised 5 Mar. 2020; accepted 22 Mar. 2020. Date of publication 14 Apr. 2020; date of current version 1 May 2020. (Corresponding author: Laurens Versluis.) Recommended for acceptance by T. Kosar. Digital Object Identifier no. 10.1109/TPDS.2020.2984821

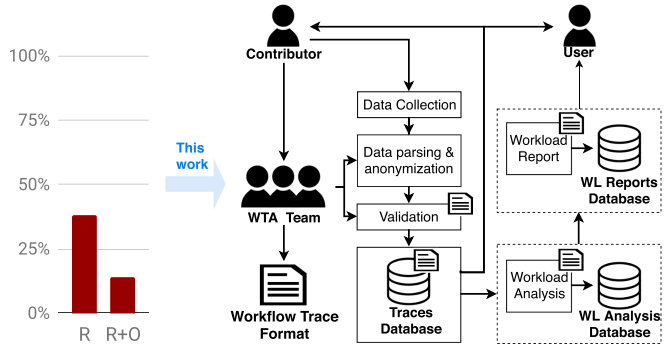

Fig. 1. A visual map to this work: (*left*) The problem: infrequent use of Realistic ($\approx 40\%$) and Open-source ($\approx 15\%$) workflow-traces in representative articles (see Section 2), which can affect the relevance and reproducibility of experiments for the entire community. (*right*) Toward an answer: the WTA stakeholders, process, and tools provide the community with open-source traces of relevant workflows running in public and private computing infrastructures.

downtime or the operational costs required. Instead, workflow traces can be replayed in silico, allowing multiple setups to run in parallel, testing individual components, etc. without the downtime nor costs. Although *realistic workflow traces* are key for testing, tuning, validating, and inspiring system designs, they are currently still scarce [13]. Prior work, such as WorkflowHub [14], has introduced numerous workflow traces, yet only from the science domain. As Fig. 1 (left) indicates, and Section 2 quantifies and explains, less than 40 percent of relevant articles focusing on workflow systems conduct experiments with *realistic* traces, and less than 15 percent conduct experiments with realistic and *open-source* traces.

The current scarcity of traces forces researchers to either use synthetically generated workloads or to use one of the few available traces. Synthetic traces may reduce the representatives and quality of experiments, if they do not match relevant real-world settings. Using realistic traces that correspond to a narrow application-domain may result in overfitting; Amvrosiadis *et al.* [15] demonstrate this for cluster-based infrastructures. Additionally, a lack of realistic traces may lead to limited or even wrong understanding of workflow characteristics, their performance, and their usage, which hampers the reuse of the systems tested with such (workloads of) workflows [16]. This gives rise to the research question *RQ-1: How diverse are the workflow traces currently used by the systems community?*

We identify the need to share workflow traces collected from relevant environments running relevant workloads under relevant constraints. Effective sharing requires unified trace formats, and also support for emerging and new features. For example, since the introduction of commercial clouds, clients have increasingly started to ask for better QoS, and in particular have started to increasingly express non-functional requirement (NFRs) such as availability, privacy, and security demands in traces [4], [17]. This leads us to research question *RQ-2: How to support sharing workflow traces in a common, unified format? How to support in it arbitrary NFRs?*

Persuading both academia and industry to release data is vital to address the problems stated prior. We tackle this issue with two main approaches. First, by offering tools to obscure sensitive information, while still retaining significant detail in shared traces. Second, by encouraging the same organization to share the data across its possibly multiple workflow management systems (*sources*), and by explicitly aiming to collect data across diverse application *domains* and *fields*. The availability of diverse data and tools stimulate the benefits of making available such traces, while simultaneously reducing the concerns of competitive disadvantage or of an (accidental) disclosure of sensitive information. The community is already helping with both approaches, by increasingly focusing on the problem of reproducibility. For example, ACM introduced artifact review and badges to stimulate the release of both software and data artifacts for reproducibility and verification purposes [18]. We add to this community-effort ours, which is scientific in nature: *RQ-3: What is the impact of the source and domain of a trace on the characteristics of workflows?*

Addressing research questions 1–3, our contribution is four-fold:

1) To answer RQ-1, we conduct the first comprehensive survey of how the systems community uses workflow traces (Section 2). We collect, select, and label articles from top conferences and journals covering workflow management. We analyze the types of traces used in the community, and the domains and fields covered in published studies. To improve reproducibility and promote extensions, we make public all (raw) data used for this survey.
2) To answer RQ-2, we design the WTA for open-access to traces of *workloads* of workflows (Section 3). We identify a comprehensive set of requirements for a workflow trace archive. A key conceptual contribution of the WTA is the design of a unified trace format for sharing workflows, the first to generalize NFRs support at both workflow- and task-levels. The WTA currently archives a diverse set of (1) real workflow traces collected from real-world environments, (2) realistic workflow traces used in peer-reviewed publications, and (3) workflow traces collected from simulated and emulated environments commonly used by the systems community. WTA also introduces tools to detail and compare its traces.
3) To address RQ-3, we compare key workload characteristics across traces, domains, and sources (Section 4). Our effort is the first to characterize the new trace from Alibaba, and the first to investigate the critical path task length, level of parallelism, and burstiness using the Hurst exponent on workloads of workflows. Overall, the archive comprises 96 traces, featuring more than 48 million workflows containing over 2 billion CPU core hours.
4) To validate our answers to RQs 1–3, we analyze various threats (Section 5). We conduct a trace-based simulation study and qualitative analysis. Our results for the former indicate systems should be tested with different traces to validate claims about the generality of the proposed approach.

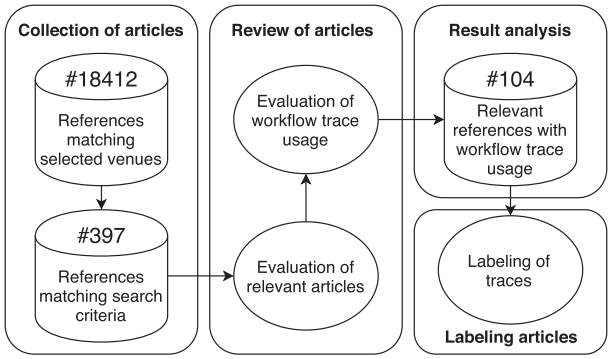

Fig. 2. The article selection process. Subsequent stages decrease the amount of articles: from a corpus of 18 412 articles, down to 104 relevant references.

2 A Survey of Workflow Trace Usage

To assess the current usage of workflow traces in the systems community and the need for a workflow archive, we systematically survey a large body of work published in top conferences and journals, and investigate articles that perform experiments using workflow traces, either through simulation or using a real-world setup. The process and outcome of this survey answer RQ-1.

2.1 Article Selection and Labeling

Selection. Fig. 2 displays our systematic approach to select articles relevant to this survey, based on [19]. First, we collect data from DBLP [20] and Semantic Scholar [21]. We filter them by venue, retaining only articles from the 10 key conferences and journals in distributed systems listed in the caption of Table 1, including TPDS. While not an exhaustive list, this covers a significant part of the systems community. This yields 18,412 articles. Next, we automatically select all articles from the last decade (2009–2018) containing the word "workflow" in either title or abstract, yielding 397 articles. This step provides articles that focus on all aspects of workflows, e.g., scheduling, analysis, and design. Finally, to obtain insights into workflow traces usage, we manually check the 397 articles. Overall, this systematic process yields 104 articles using workflow traces. To highlight the relevance of papers, we use Google Scholar to obtain citation counts. In total the 104 papers have been cited 3,965 times.

Labeling. We label for each of the 104 articles the type of trace usage. For articles explicitly describing their use, we use the label *realistic* for traces *collected* from real-world workflow executions. For all others, including workflows *extrapolated* from real-world data or *generated* from known statistical distributions, we use the label *synthetic*.

We further label traces as *open-access* (or open-source) if they are available online and to a broad audience, and *closed-access* (or closed-sources) otherwise. In our analysis, we include among the open-access traces only those that are also realistic.

We also label traces by *domain* and *field*. *Domains* are corresponding to the area of study of which the trace originates from. We label sub-domains within these domains as *fields*. We adopt the domains and fields reported by the respective authors, where mentioned. If the domain or field are not mentioned, yet the application appears in another article by name and with labels, we remain consistent in our labeling by adopting the domain/field from this prior article. We have not encountered cases where an application is labelled as belonging to multiple domains or fields. We identify in articles explicit use of traces from the domains "scientific", "engineering", "multimedia", "governmental", and "industry", and from fields such as "bioinformatics", "astronomy", "physics", etc. We further label a trace with *uncategorized* when its origin remains unexplained.

All data used in this survey is available as open-access data[1] and can be used to verify and extend this survey.

2.2 Types of Traces Used in the Community

We analyze here the types of traces used by the community, with the following Observations (Os):

O-1: Less than 40 percent of articles use realistic traces.
O-2: Only one-seventh of all articles use open-access traces.

Table 1 presents the types of traces used in the community, focusing on realistic (R) and open-access (R+O) traces. The community uses traces for experiments across both conference and journal articles, across various levels of (high) quality. In contrast to this positive finding, the results indicate that, from the total number of articles using traces at all, the fraction of articles using realistic and even open-access traces is relatively small. Across all venues, only 38 percent of the articles use at least one realistic trace, and only 13 percent of the articles use at least one open-access trace.

These findings match the perceived difficulty in reproducing studies in the field [11], [12], and may hint why so few of these seemingly successful designs are adopted for use in practice [22].

2.3 Workflow Domains and Fields

We analyze the domains and fields from which the community sources workflows, with as main observations:

O-3: The community sources workflows from 5+ domains and 25+ fields.
O-4: Traces containing scientific workflows are used significantly more (20x) than workflows from other domains, e.g., industry and engineering, in the surveyed articles.
O-5: Bioinformatics workflows are the most commonly used, but three other fields exhibit usage within a factor of 3.
O-6: Many traces have uncategorized domain (14 percent) and/or field (31 percent).

Overall, we find that the community uses diverse workflows, sourced from 5+ domains and 25+ fields.

We further investigate the distribution of use, per domain and per field. Fig. 3 (top) shows that the scientific domain is over-represented in the literature in the top-five trace domains encountered, due to the large number of available open-access traces and from their conventional use in prior work. In particular, a large portion of the articles use workflow traces from the Pegasus project, which covers the scientific domain. The number of traces in this domain exceeds 200, which is larger than the number of

1. https://github.com/atlarge-research/wta-analysis

TABLE 1
Workflow Trace Usage in Venues Having at Least One Paper Returned in the Initial Query

	Acronym	Total	FGCS	CCGrid	*TPDS*	Other
T	Articles using traces	104	37	17	17	33
R	Articles using *realistic* traces	40 (38%)	13 (35%)	8 (47%)	6 (35%)	13 (39%)
R+O	Articles using traces that are both *realistic* and *open-access*	14 (13%)	6 (16%)	2 (12%)	3 (18%)	3 (9%)

The venues with > 5 hits have their individual column. The column "Other" shows combined results for conferences with ≤ 5 hits: ATC, CLOUD, CLUSTER, e-Science, Euro-Par, GRID, HPDC, JSSPP, IC2E, ICDCS, ICPE, IPDPS, NSDI, OSDI, SC, SIGMETRICS, and WORKS. Percentages are computed from the total in the corresponding column, e.g., 13 out of 37 for the cell corresponding to row R and column FGCS.

articles in the study as each article uses multiple traces. In contrast, the next-largest domains are industry and engineering, each with less than 10 traces representing less than one-twentieth of the scientific domain.

We remark the positive diversity of the workflow domains, considering that the entire community is tempered by the extreme focus on scientific workflows. This confirms the bias demonstrated by Amvrosiadis *et al.* [23] with the popular Google-cluster traces. A similar situation appears for fields, but more tempered, as Fig. 3 (bottom) indicates. A large portion of the traces have their domains and fields as "uncategorized" (14 and 31 percent, respectively) which is unhelpful when determining if the proposed solution works in a certain environment.

Overall, the results reveal that the community has a strong bias for one domain (scientific) and favors scientific fields (especially bioinformatics). We conjecture the large amount of open-access data from these fields facilitates this bias. This is consistent with our findings **O-4** and **O-5**, and with the assumption of people selecting traces with equal probability. An alternative is that the domains and fields whose data are used more, share artifacts that are more easily reused and rerun. An example of a well-known initiative for reproducibility in the scientific domain is the MyExperiment repository [24]. To overcome such biases, and to further reduce the large fraction of uncategorized traces evident in both plots of Fig. 3, we posit the community should require that open-access and *diverse* traces be used in articles claiming the generality of their techniques and indicate the domains and fields of the workflows used.

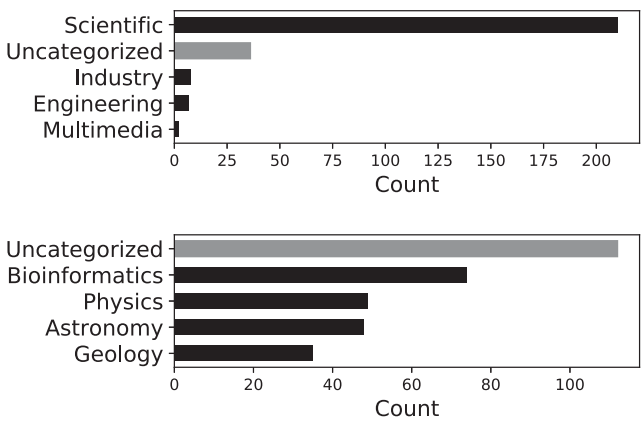

Fig. 3. (*top*) Top-5 (out of 6) domains and (*bottom*) Top-5 (out of 28) fields from which the community sources workflows. ("Uncategorized" for unclear domain or field.)

3 The Workflow Trace Archive

In this section, we outline the design of the WTA, the unified trace format used, tools to support consumers with the trace selection according to their use-case, and give a summarized overview of the current contents of the archive. Furthermore, that facilitates the continuous growth of the archive, we provide tools for trace anonymization and a collection of trace parse scripts for different trace sources.

Similar to how the design of experiments is now commonly described in publications in our field, as the setup leading to experimental results, we include an overview of the design process that led to the design presented in this section. Outlining the design and the process that led to the design is important for understanding how the final design came to be and how it fits the intended goal [25].

We started by listing initial requirements (see Section 3.1) that the WTA has to fulfill, and *co-evolved the requirements with the development of the solution* (the archive). For example, we added explicitly the requirement to provide scripts and datasets to aid users in building their own tools, as we discovered how difficult it was to engineer them from scratch (see Section 3.6). Next, we defined an initial format, centered around a number of unique features, such as non-functional requirements (NFRs) that are missing in other workflow trace formats. We improved this format iteratively, to meet the requirements and/or to pass various thought experiments. For the latter, whenever we encountered a new data-format that was not fully covered by our format, we discussed which properties and/or objects should be added to the format (see Section 3.4). We assessed the trade-off between format comprehensiveness (what to include?) and brevity (what is too much or too complex?) based on personal experience, on the perceived importance of data-fields in literature, and on their frequency of use in other archives. Finally, we designed the analysis tools iteratively, including in them initially our own ideas and then aspects highlighted by other archives, literature reports, and perceived shortcomings.

3.1 Use Cases and Requirements

We foresee four direct use cases for the WTA. First, trace characterization and workload analysis for understanding and tuning systems. As workloads evolve it is important to characterize the changes in, e.g., structure and resource consumption to see if schedulers require change, can be improved or if these changes can be exploited. Such characterizations can provide interesting insights (see Section 4).

Second, experimentation using emulation or simulation. As discussed in Section 1, emulations and simulations may

be the only viable option for specific scenarios (e.g., what if?, long-term operational analysis). Having an archive that offers diverse, heterogeneous traces allows for more diverse testing scenarios. Especially when a new scheduler is developed for multiple domains or scenarios, it is important to experiment with diverse workloads covering the scenarios and domains targeted (see Section 5, **C-1**).

Third, workload and operational models can arise from the characterization and simulation results. In turn, these models can lead to new insights or to new variations to experiment with.

Last, such data can be used for education and training. As systems grow more complex, education and training becomes more important for both students and employees [26]. Models and heterogeneous traces are useful in education, to demonstrate scenarios and to provide hands-on experience.

To meet these use cases, we identify five key requirements for the structure, content, and operation of a useful archive for workflow traces.

R-1*: Diverse Traces for Academia, Industry, and Education.* Trace archives, such as Google's and Alibaba's, offer only workloads from a single domain, e.g., industrial workloads.

We identify as requirement that an archive must include a diverse set of traces to cover a broad spectrum of workflow sizes, structures, and other characteristics, including both general characteristics to many domains and fields, and idiosyncratic characteristics corresponding to only one domain or field. This requirement is based on the conjectures that different traces can have workflows with significantly different characteristics (tested in Section 4) and such differences impact system performance (tested in Section 5, **C-1**).

Addressing this requirement is important for academia to demonstrate the generality and applicability of a novel approach, for industry to test production-ready systems or to validate techniques proposed by academia [27], and for education to train employees on more complex systems.

R-2: A Unified Format for Workload of Workflows Traces.

To improve the reusability of diverse traces and to support the reproducibility of experimental results, long-term, we identify as a requirement the use of a unified trace format for workloads of workflows. The format must cover a broad set of data about the workloads and about the workflow management systems including: workload metadata; workflow-level data including NFRs; task-level data including per-task NFRs and operational metadata; inter-dependencies between tasks and other operational elements such as data transfers; system-level information including resource provisioning, allocation, and consumption; etc.

Addressing this requirement simplifies trace exchange and integration effort, prevents redundant work for other users, and supports the development of dataset independent tools (expressed as **R-3**).

R-3*: User level adapted insights into Trace Properties.*

To improve trace discovery, the archive must provide detailed trace insights adapted to the level of the broad audience, from beginner to expert, as implied by **R-1**. Broad insights include *extrinsic properties*, such the number of workflows and tasks, and *intrinsic properties*, such the workflow arrival patterns and the resource consumption per-task. In contrast, detailed expert-level insights include *analysis of single traces* at workload-, workflow-, and system-level; and *collective analysis* across all traces or traces filtered by a feature (e.g., all traces of a domain or field). These properties must be accessible through readily available tools (see **R-4**) and, possibly, through interactive online reports. Addressing this requirement helps to correlate information across different traces, resulting in better quantitative evidence, intuition about otherwise black-box applications, and understanding that helps avoiding common pitfalls [28].

R-4*: Tools for Trace Access, Parsing, Analysis, Validation.*

The most important tool is the online presence of the archive itself. The archive must further provide tools to parse traces from different sources to the unified format (see also **R-2**), to provide insight into traces (see also **R-3**), and to *validate* common properties (e.g., the presence of and correctness of properties). An absence of such tools would lead to users unable to select appropriate traces, validate their properties, and compare them.

The archive should further aid users in building more sophisticated tools. Newly built tools can then be added to the selection of tools so more parties can make use of them (contributing to **R-5**)

R-5*: Methods for Contribution.*

The archive must reflect the continuous evolution of workflow use in practice, by increasing the coverage of different scenarios. We make a distinction between two types of contribution: (1) traces from a new domain or application-field, and (2) traces, introducing new properties. To facilitate the former contribution, the archive must provide a method for the upload and (basic) automated traces verification. To facilitate the latter, the format must integrate specific provisions that enable upgrades and long-term maintainability, such as adding a version to each component of the format.

Addressing this requirement encourages new and existing contributors to submit new traces. In particular, tools to add new domains are of particular importance, to support emerging paradigms with realistic data.

3.2 Overview of the WTA

We design the WTA as a process and set of tools helping a diverse set of stakeholders. We consider three roles for the WTA community members, outlined in Fig. 1. The *contributor* supplies, as the legal owner or representative, one or more traces to the WTA. A workflow trace contains historical task execution data, resource usage, NFRs, resource description, inputs and outputs, etc. To fulfill **R-5**, the *WTA team* assists the contributor in parsing, anonymizing, and converting the traces into the unified format (Section 3.4), minimizing the risk of competitive disadvantage, and verifying their integrity. WTA fulfills **R-1** as it incrementally expands with contributors of traces from different domains with different properties.

The *user* represents non-expert or expert trace consumers. Non-expert users often need to rely on generic domain or trace properties, whereas the expert users have detailed knowledge of their system and require fine-grained details for selecting the correct trace. In addition, expert users may comment on (missing) properties and may develop new tools, models or other techniques to further compare and rank the traces. Both user types require assistance in selecting

TABLE 2
Overview of the Current WTA Content, Grouped by Source

Source ID. Name	#WL	D	DS	#PA	#PL	#S	#A	#WF	#T	#U	#G	Year(s)	Timespan	TCH
S1. Askalon Old	2	Eng	-	-	1	-	mixed	4,583	167,677	*7	*6	2007	19 months	4,685,300
S2. Askalon New	67	Sci	-	*2	2	67	*3	1,835	91,599	*67	*67	2016	47 days	193
S3. LANL	2	Sci	-	-	1	-	mixed	1,988,397	475,555,927	-	-	2011-2016	63 months	*9,625,431
S4. *Pegasus*	8	Sci	-	-	*6	-	8	56	10,573	9	-	2011	4 days	1,477
S5. Shell	1	Ind	-	-	1	-	mixed	3,403	10,208	-	-	2016	10 minutes	25
S6. SPEC	2	Sci	-	-	1	-	mixed	400	28,506	-	-	2017	-	1,231
S7. Two Sigma ‡	2	Ind	-	-	1	-	mixed	41,607,237	50,518,481	610	1	2016	16 months	69,992,196
S8. *WorkflowHub*	10	Sci	*5	*4	5	-	3	10	14,275	10	-	2017	-	52
S9. *Alibaba*	1	Ind	-	-	1	-	mixed	4,210,365	1,356,691,136	1	1	2018	8 days	1,526,925,484
S10. Google	1	Ind	-	1	1	-	mixed	494,179	17,810,002	430	1	2011	29 days	434,821,345
Total	96	-	*5	*7	*20	67	-	48,310,465	1,900,898,384	*1,134	*76	-	-	2,046,052,734

*Legend: D = Domain, DS = Datasets, PA = parameters, PL = Platform, S = Setup, A = Applications, WL = workload, WF=workflow, T = task, U = user, G = group, * = minimum, Eng = Engineering, Sci = Scientific, Ind = Industry, and TCH = Total Core Hours. Items in bold are workloads introduced by this work. Items where workflows are for the first time analyzed in this work are in* italics. *The symbol ‡ next to S7 indicates data with promise to release, but for which the legal forms have not been completed yet; WTA can already release all other workloads.*

the most suitable trace given a set of criteria (Section 3.5) as well as analysis and validation (Section 3.6) from the available set of traces (Section 3.7). To support both user types, the WTA discloses both high-level and low-level details.

3.3 Workflow Model

There are numerous types of workflow models used across different communities. A 2018 study by Versluis *et al.* finds DAGs are the most commonly used formalism in computer system conferences [29]. Popular formalisms such as CWL [30] and Condor DAG [31] are also DAG-based. Therefore, for the first design of this archive, we adopt DAGs as the workflow model.

A workflow constructed as a DAG in which nodes are computational tasks and directed edges depict the computational or data constraints between tasks. Entry tasks are tasks with no incoming dependencies and, once submitted to the system, immediately are eligible for execution. Similarly, end tasks are nodes that have no outgoing edges. A collection of workflows submitted to the same infrastructure over a certain period of time is considered a *workload*.

Although popular, we specifically do not focus in this work on BPMN and BPEL, Petri nets, hyper graphs, general undirected, or cyclic graphs. These formalisms either include business and human-in-the-loop elements [32] or add additional complexity due to having a large set of control structures such as loops, conditions, etc. [9] which we consider out of scope for this work.

Executable formalisms are meant to define what resources and software should be available *before* execution. Our formalism needs to capture the system state *during* execution. Both types of formalisms are needed, and are complementary to each other. For example, a person could use the CWL to define and run their workload, then turn to our formalism and tools to analyze its execution and subsequently improve various operational aspects.

Given the different nature of these formalisms, if we were to extend an existing executable workflow formalism, e.g., CWL, several elements would not be used. This would lead to *feature creep*. Conversely, the additions made by our formalism could be regarded as feature-creep by the CWL community. This is emphasized by the CWL community currently developing CWLProv [33]. This formalism aims at fully reproducible workflows, including re-execution which is not a goal of the WTA. While promising, CWLProv is still a work in progress; elements such as capturing resource usage (e.g., CPUs and power consumption) are still lacking.

3.4 Unified Trace Format

Creating a unified format (**R-2**) requires from the designer a careful balance between limiting the number of recorded fields while supporting a diverse set usage scenarios for all stakeholders in Section 3.2. Modern logging and tracing infrastructure can capture thousands of metrics for each machine and workflow-task involved [34], from which the designer must select. We specifically envision support for common system and workflow properties found in the typical scenarios considered in the top venues surveyed in Section 2, such as engineering a workflow engine [35], characterizing the properties of workloads of workflows [36], and designing and tuning new workflow schedulers [37].

Our unified format attempts to cover different trace *domains*, while preserving valuable information, such as resource consumption and NFRs, contributing to fulfilling **R-1 and 3**. The full technical description of the format can be found in our technical report [38] and on the WTA website.[2] By analyzing the raw data formats, we carefully selected useful properties to include in our unified format, omitting low-level details, such as cycles per instruction, page cache sizes, etc.

Answering RQ-2 and fulfilling **R-2**, our trace format is the first to support arbitrary NFRs both at task and workflow levels. For example, one of the LANL traces (introduced in Table 2) contains deadlines per workflow and the Google cluster data features task priorities, both are supported by the WTA unified format. Capturing these properties is important to test QoS-aware schedulers.

As depicted in Fig. 4, the WTA format includes seven objects: *Workload*, *Workflow*, *Task*, *TaskState Resource*, *ResourceState*, and *DataTransfer*. Each of these objects contains a version field, updated whenever the set of properties is altered (**R-5**).

2. https://wta.atlarge-research.com/traceformat.html

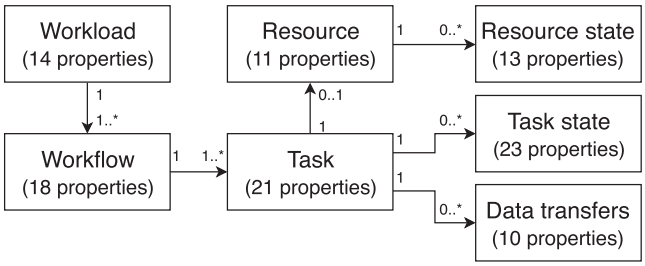

Fig. 4. The WTA trace format.

Each trace is a single *workload*, consisting of multiple workflows and their *arrival process*. Workload properties include the number of workflows, tasks, users, domain and field when available, authors list, and resource consumption statistics. Each workload belongs to one or more *domains*. and contains a *description* including its source, execution environment, etc.

Each *workflow* in the workload has a unique identifier, an arrival time, and contains a set of tasks and several properties, including scheduler used, number of tasks, critical path length, NFRs, and resource consumption. Each workflow also has the *name* of its *field* of study, when possible. Different related fields constitute a domain.

Each *Task* has a unique identifier and lists its submission and waiting time, runtime and resource requirements, including required (compute) resource type, memory, network, and energy usage. Additionally, each task provides optional dictionaries for task-specific execution parameters and NFRs. To model dependencies between tasks, the WTA format maintains for each workflow its topology by specifying parents and children per task. Similarly, data dependencies are recorded as a list of data transfers.

Resource objects cover various resource types, such as cloud instances, cluster nodes, and IoT devices. A resource has a unique identifier and contains several properties, such as resource type (e.g., CPU, GPU, threads), number, processor model, memory, disk space, and operating system. An optional dictionary provides further details, such as instance type or Cloud provider. The *ResourceState* event snapshots periodically the resource state, including availability and utilization. Analogous to the ResourceState, the *TaskState* records periodically the resource consumption of the task (the Task object records the resource demand).

Each *DataTransfer* describes a file transfer from a source to a destination task, which can be a local copy on the same resource or a network transfer from a remote source, etc. To support bandwidth analysis, a data transfer introduces submission time, transfer time, and data size. Each data transfer also provides an optional dictionary with detailed event timestamps (e.g., pause, retry).

3.5 Mechanisms for Trace Selection

We address **R-3** by assisting archive users in retrieving appropriate traces for their scenarios, using filter and selection mechanisms. The website is the most important such filter and mechanism, containing an overview of all traces in a general table with the number of workflows, tasks, users, etc. This table is sortable and searchable, allowing website users to interact with the more than 90 traces currently in the WTA (column "#WL", row "Total" in Table 2).

TABLE 3
Trace Anonymization Methods Used in WTA Tools

Obfuscation method	Description
IP	Encodes IPv4 addresses
Mail and host	Obfuscate mail and host names
File paths	Hide file paths in Linux and Windows format
Executable files	Encode executable file names, e.g., py, sh, exe, jar
All files	Hide all file names, ending with 2, 3, or 4 letters
Keywords	Anonymize a list of custom keywords
All	Apply all obfuscation methods listed above

We provide, online and as separate tools, a detailed report for each trace. Each report includes automatically generated statistics, such as the number of workflows and tasks, then resource properties such as compute, memory, and IO, and job and task arrival times and runtime distributions (see Section 4). The metrics featured in the report are reported as important by prior studies [39], [40] and enable developers to select traces appropriate for their intended use-case.

3.6 Tools for Analysis and Validation

We implement the unified trace format using the Parquet file format and the Snappy compression algorithm. Parquet is a binary file format that is supported by many big data tools such as Apache Spark, Flink, Druid, and Hadoop [41]. Many programming languages also have libraries to parse this format, such as PyArrow for Python and parquet-avro for Java. Snappy[3] compression reduces the size of the dataset significantly and has low CPU usage during extraction.

Beside trace selection support and to address **R-4**, the WTA offers several tools to facilitate and incentivize the continuous growth of the archive. Most of these tools required significant engineering effort to develop, due to the typical challenges of big data processing (high volume, noisy data, diverse input-formats, etc.). The WTA simplifies the upload of new traces by providing a set of parsing scripts for different trace sources, such as Google, Pegasus, and Alibaba. Parsing traces can become non-trivial, once they grow both in complexity and size. Such traces require big data tools, such as Apache Spark, and enough resources, a cluster, to compute. Noisy data raise another non-trivial issue: both Google's and Alibaba's cluster data contained either anomalous fields, undocumented attributes, and non-DAG workflows. Some of these issues were never discovered by their respective communities and were corrected in our parsing tools. Debugging, filtering, and correcting noisy big data requires significant compute power and detailed engineering.

Because traces may contain sensitive information, the WTA offers a *trace anonymization tool*, which supports users to automatically replace privacy and security-related information, to avoid an accidental reveal of proprietary information. Specifically, to remove sensitive information from trace files, we use two common techniques [42], *culling* and *transforming*. Culling is done during trace conversion, by omitting parts of the raw trace data which do not match our workflow trace format. For the transformation, as presented in Table 3, our anonymization tool automatically scans the

3. https://github.com/google/snappy

TABLE 4
Overview of Properties Available Per Source

Source ID	Task details	Task resource req.	Structural information	Disk	Memory	Network	Energy	NFRs
S1	✓	✓	✓					
S2	✓	✓	✓					
S3	∼	✓	∼					✓
S4	✓	✓	✓					
S5	✓	✓	✓					
S6	✓	✓	✓			✓		
S7	✓	✓	∼		✓			
S8	✓	✓	✓		✓	✓		
S9	✓	✓	✓	✓	✓	✓		
S10		✓	∼	✓	✓			✓

Legend: ✓ = available, ∼ = partially available, blank = not available, and Task details = individual task information.

workflow trace file for sensitive data, such as IP addresses, file paths, names, etc., by string pattern matching. Beside these standard sensitive-data checks, the WTA offers the option to search for custom privacy-critical strings.

Finally, all matched strings are replaced by a salted SHA-256 hash key. This approach using cryptographic hash functions offers protection of sensitive data, while preserving the relationships between the matched values in the same trace file [42]. Additionally, our tool hides potential relations to other trace files by adding a salt of length 16 to the hash key generation, which is randomly generated on each tool run.

To validate traces, the WTA provides a *validation* script that checks the integrity and summarizes important characteristics of a trace. During trace conversion, using the validation script, we successfully identified several parse bugs and inconsistencies in the data that we subsequently corrected.

Specifically, because tasks build the base of each trace, our tool checks if all contained tasks are well defined. This, for example, means that all parsed control dependencies, such as children and parents, link only to existing tasks with valid properties. A task property is valid, if the parsed property type matches the property type definition, and the property value is allowed e.g., task runtime > 0. Based on and similar to this fundamental validation, our tool provides options to check the workflow and data transfer properties to identify inconsistencies, as well.

These tools help combating perceived barriers to share data described by Sayogo *et al.* [43]. Several technological barriers are addressed by using a unified format and validation (data architecture, quality, and standardization), Legal and policy barriers are more difficult to address. Our anonymization tool aids in overcoming the data protection barrier, yet legal and other enforced policies may require tailored solutions.

Besides offering these tools, the WTA also hosts the trace data, addressing logistic and economical barriers. The increasing focus on sharing data artifacts by the community, is lowering the barrier regarding competition for merit and reputation for quality and bolsters the culture of open sharing. Finally, each trace has its own DOI by also uploading it to Zenodo[4] which can be cited and thus provides authors with the appropriate credits (incentive barrier).

3.7 Current Content

Having a diverse set of traces available is necessary to use in experimentation. When using traces in experimentation, different traces should be used to prove generality of the proposed approach (see Section 5). Gathering and parsing raw logs and other traces requires significant computing effort. Using 16 nodes (32 eight-core Xeon E5-2630 v3 and 1TB RAM) from the Dutch DAS5 super computer [44], several traces require up to a day to compute using big data tools such as Apache Spark. In total, the WTA team spent more than two person months on converting traces to the unified trace format. By offering these parse scripts and the data, we contribute to **R-4**.

The WTA features currently 96 workloads from 10 different sources, with over 48 million workflows and 2 billion CPU core hours. All of them are available on the WTA archive website.[5] Each workload is uniquely identified by a combination of the following properties if available: source, runtime environment, application, and application parameters [45]. Tables 2 and 4 summarize these traces. From these tables we observe that WTA contains a vast amount of different traces, from different sources and domains, with various number of workflows, properties, number of tasks, timespans, and core hour counts. Although supported by our format, no trace currently has information on energy consumption, highlighting the need of such traces [14]. These traces are collected by combining open-access data (logs, traces, etc.) and closed-access data throughout the years in collaboration with both industry and academia. This contributes to **R-1**.

This diversity enables new workflow management techniques and systems to be thoroughly tested for their feasibility, strengths, and, equally important, weaknesses.

4 A Characterization of Workloads of Workflows

To answer RQ-3, we perform in this section a characterization of the workloads in the WTA. These workloads originate from publicly available archives combined with workloads we obtained from collaborations. As we expect these workloads to be heterogeneous in many dimensions, we characterize them using a variety of metrics and properties, including workflow size, resource usage, and structural

4. https://zenodo.org/

5. https://wta.atlarge-research.com/

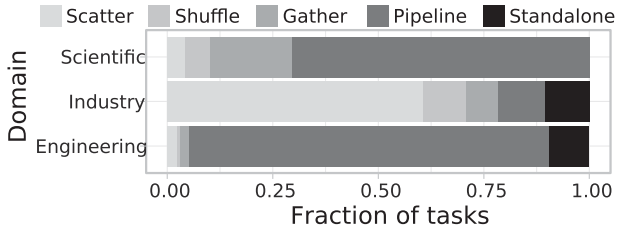

Fig. 5. Structural workflow patterns, per domain.

patterns. Our characterization reveals significant differences between workloads from different domains and sources. Such differences further support our claim that the community needs to look beyond just scientific workloads, and consider a wider range of domains and sources for experimental studies when developing workflow management systems aimed at multiple domains or for general applicability.

We present in this section only detailed insights that lead to new observations for the community. We include in our technical report other types of analysis, such as task and workflow inter-arrival times, task and workflow runtimes, and their breakdown per domain and source [38].

4.1 Structural Patterns

O-7: Scientific, industrial, and engineering workflows exhibit various structural patterns, but at least 60 percent of tasks in a domain match the dominant pattern of that domain.

O-8: Industry workflows stand out by exhibiting primarily scatter patterns, as opposed to pipeline operations.

This characterization quantifies five structural patterns in workflows often used by researchers [46]: scatter (data distribution), shuffle (data redistribution), gather (data aggregation), pipeline, and standalone (process). Investigating these structural patterns is important to understand the types of applications being executed and tune a system's performance. We exclude from this analysis the LANL, Two Sigma, and Google traces, which lack structural information, that is, task parent-child relationship information.

Fig. 5 depicts the structural patterns found per domain. From this figure, we observe that in each domain a dominant pattern emerges that accounts for 61–85 percent of tasks. In the scientific and engineering domains, the majority of tasks are simple pipelines. Interestingly, the industrial workflows include primarily scatter operations. This observation matches known properties of the Alibaba trace, which accounts for over 99 percent of tasks with structural information we analyzed in this domain. In particular, the Alibaba trace includes MapReduce jobs, each consisting of many "map" tasks (scatter operations) and a smaller number of "reduce" tasks (gather operations).

4.2 Arrival Patterns

O-9: From all domains, industrial traces show on average orders of magnitude higher rates of task arrival.

O-10: Scientific traces can show high variability in task arrival rates, unlike industrial and engineering traces.

O-11: Two Sigma shows a typical workday diurnal pattern.

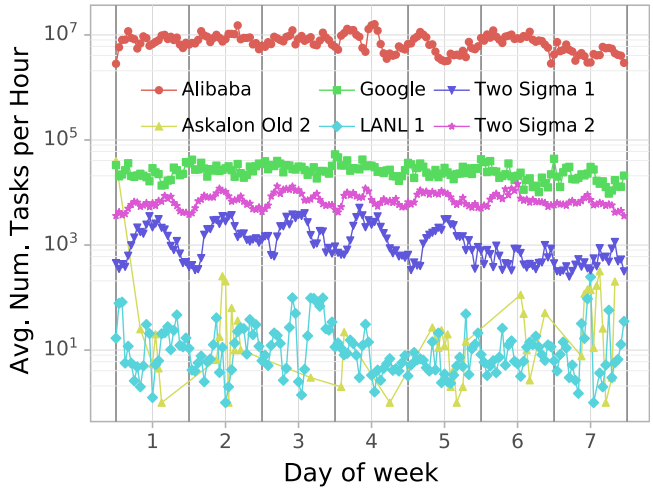

Fig. 6. Daily task-arrival trend, per source.

To investigate the weekly trends that may appear in workload traces, we depicts in Fig. 6 for several traces the average number of tasks that arrive per day of the week. We omit the Askalon new source from the hourly task-arrival plot as they contain 4 or 5 data points, which is too few to plot a trend. We observe that traces have significantly different arrival rates and patterns. The Alibaba trace features the highest task arrival rates, peaking at over 10,000,000 tasks per hour. Google and the Two Sigma workloads follow with 100-10,000 tasks per hour. This shows that industrial workloads included in this work have significantly more tasks per hour than the other compute environments, which agrees with companies such as Alibaba and Google operating at a global scale. The non-industrial traces show significant fluctuations throughout the week, whereas both Alibaba and Google do not. This might be due to the global, around-the-clock operation of Alibaba's and Google's services, which can lead to a more stable task arrival rate.

To observe differences in daily trends, we depict the average task rate per hour of day in Fig. 7. This figure reaffirms our observation that the two largest traces–Alibaba and Google–have a relatively stable arrival pattern throughout the day. In contrast, the Two Sigma 1 trace exhibits a

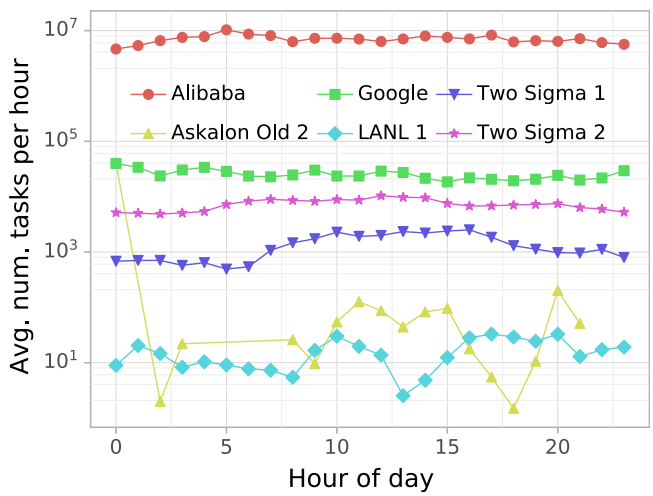

Fig. 7. Hourly task-arrival trend, per source.

TABLE 5
The Design and Setup of our Characterization

ID	Section	Description	Traces	Metric	Granularity
E1	4.1	Analyze structural patterns in workflows per domain	All but S3, 7, 10	Structural patterns	Workflow level
E2	4.2	Longitudinal analysis	S1, S3, S7, S9, S10	Tasks per day	Workload level
E3	4.3	Analysis of burstiness per trace	All but S4-8	Hurst exponent	Workload level
E4	4.4	Measure the level of parallelism per workflow	All but S3, 7, 10	Level of parallelism	Workflow level
E5	4.5	Analysis of critical path length	All but S3, 7, 10	Critical path length	Workflow level

typical office hours pattern; task arrival rates increase around hour 7 and start dropping around 17. The same pattern occurs to a lesser extent in the Two Sigma 2 trace. The highly variable arrival rates of tasks in the LANL traces, as observed in Fig. 6, are also evident in our analysis of daily trends. We study this in more depth in Section 4.3.

4.3 Burstiness

O-12: Most traces investigated exhibit bursty behavior within small window sizes.
O-13: The LANL trace exhibits maximum burstiness at medium window sizes.
O-14: The largest traces (Alibaba and Google) exhibit uniquely bursty behavior: low burstiness at small and high burstiness at large, window sizes.

To investigate if workloads expose bursty behavior, a special kind of arrival pattern, the Hurst exponent H is used. H quantifies the effect previous values have on the present value of the time series. A value of $H < 0.5$ indicates a tendency of a series moving in the opposite direction based on the previous values, and thus exhibit jittery behavior (sporadic burst). A value of $H > 0.5$ indicates a tendency to move in the same direction, and thus towards well defined peaks (sustained burst). When $H = 0.5$, the series behaves like a random Brownian motion.

In this experiment, we inspect busty behavior by computing the Hurst exponent for task arrivals. The results of this experiment are visible in Fig. 8. From this figure, we observe most traces depict bursty behavior at least for one of small, medium, and large window size. They are also not bursty for at least one window size. This is expected, as in most systems task arrivals vary at (sub-)second interval. Interestingly, LANL traces exhibit most bursty behavior at medium window sizes. This might be due to national laboratories workflows being submitted in batches. A batch of tasks is submitted all at once, leading to a burst. But, the batch itself is processed at a constant rate. The workload is also stable over longer time periods as evidenced by $H \approx 0.5$ for larger windows. Finally, the two largest traces in this work, Alibaba and Google, exhibit increasingly burst behavior for larger windows. This indicates that for larger arrival times, the workloads (in absolute numbers) vary more than for the other sources. This matches the observations in Section 4.2.

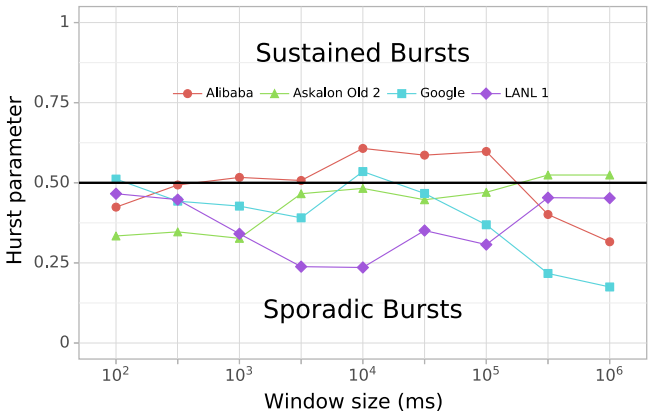

Fig. 8. Hurst exponent estimations for different time windows per trace. Horizontal axis does not start at zero.

4.4 Parallelism in Workflows

O-15: Task parallelism per workflow can differ significantly between workload domains and sources.
O-16: Industrial workflows exhibit the highest level of parallelism.
O-17: Out of all sources, Alibaba workflows have the highest level of parallelism, followed by Pegasus and WorkflowHub.

With the structural patterns observed, we investigate if the large occurrence of the pass-through patterns expresses in a high level of parallelism. The level of parallelism indicates how many tasks can maximally run in parallel for a given workflow, provided sufficient resources. Fig. 9 depicts the approximated level of parallelism per domain. The approximation algorithm used produces results very close to the true level of parallelism as demonstrated by Ilyushkin *et al.* [47]. From this figure, we observe the industrial domain exhibits the highest level of 99th percentile parallelism, up to hundreds of thousands of tasks. This is likely a consequence of the many MapReduce workflows, which are highly-parallel by nature, that are present in the Alibaba trace. Alibaba also contains bag of tasks workflows, which by nature have a high parallelism. Scientific workflows exhibit low median parallelism but high 99th percentile parallelism, featuring levels of parallelism up to thousands of tasks. Engineering traces exhibit a moderate amount of

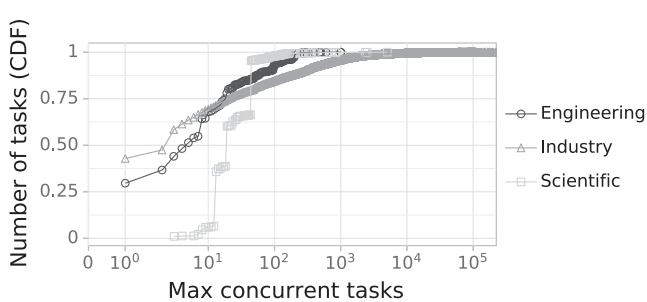

Fig. 9. Workflow level-of-parallelism, per domain.

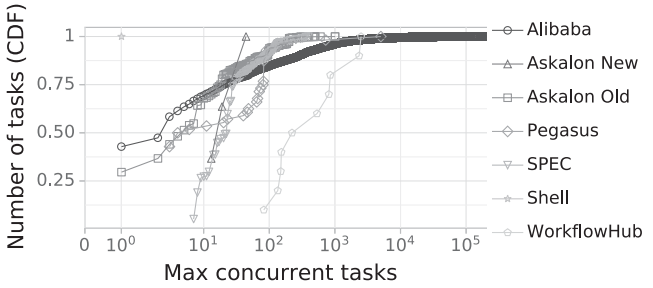

Fig. 10. Workflow level-of-parallelism, per source. Curves are shaded by domain, to further reveal patterns.

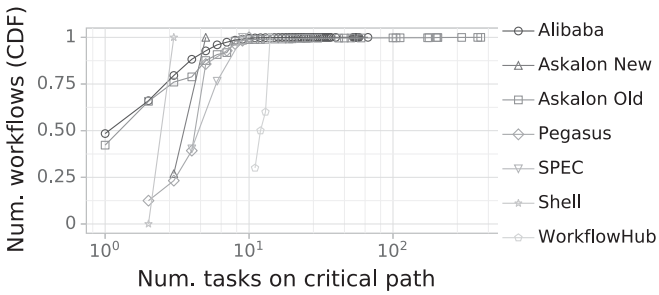

Fig. 12. Workflow level-of-parallelism, per source. Curves are shaded by domain, to further reveal patterns.

median parallelism, between industry and scientific, with at most 1000 concurrent running tasks.

Fig. 10 shows the level-of-parallelism per source. From this figure, we observe that Alibaba exhibits the highest levels of parallelism, as discussed previously. Second are the Pegasus and WorkflowHub workflows. These sources contain a variety of scientific applications, commonly known for their parallel structures, as observed in Section 4.1. Other traces demonstrate less parallelism, with up to 1000 concurrent running tasks. As Shell exist entirely of sequential pipelines, the source does not exhibit any variation.

4.5 Limits to Parallelism in Workflows

O-18: Workflows from the scientific domain have significantly different critical-path lengths.
O-19: The amount of tasks on the critical path is the highest for engineering workflows.
O-20: Although highly parallel, industrial workflows exhibit longer critical paths than scientific workflows.

The critical path (CP) refers to the longest sequence of dependent tasks in a workflow, from any entry task to any exit task. By quantifying the CP length, we investigate if workflow runtimes are primarily dominated by a few heavy tasks, or by many small tasks. Fig. 11 presents the results of this characterization per workload domain. From this figure we observe the CP length for engineering workflows is the highest. This matches with the parallelism observations in Sections 4.1 and 4.4. Interestingly, even though industrial workflows are often highly parallel, their critical paths are often longer than those of scientific workflows. This indicates that industrial workflows are bigger in size than scientific workflows, which our data supports.

Fig. 12 presents the results of CP characterization per workload source. From this figure we observe the CP length differs significantly per trace. Based on the prior findings, the engineering traces are expected to show longer critical paths. As we can observe, the Askalon old traces contains workflows with the longest critical path. Alibaba workflows also exhibit long critical paths, indicating their workflows next to being highly parallel, also contain a lot of tasks with stages. More concentrated, the other traces exhibit lower critical path lengths, yet the traces are still clearly distinct. As the Shell trace contains solely sequential workflows, the critical path length is one.

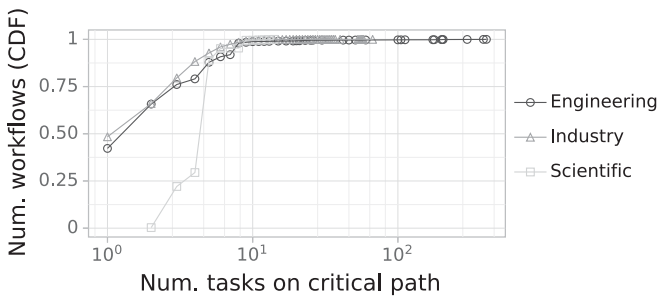

Fig. 11. Workflow level-of-parallelism, per domain.

5 Addressing Challenges of Validity

In this section, we discuss challenges to the validity of this work. We address the challenges through either trace-based simulation (the first) or argumentation (the others).

Challenge **C-1. Trace diversity does not impact the performance of workflow schedulers.** As outlined in Sections 3.7 and 4, the WTA traces are diverse. However, *what is the impact of trace diversity?*

To demonstrate the impact of trace diversity on scheduler performance, we conduct a trace-based simulation study. The simulator used is an optimized version of DGSim [48] which we publish as open-access artifact.[6] We simulate workloads from five sources using two scheduler configurations. We equip the simulated scheduler with either the first-come first-serve (FCFS) or the shortest job first (SJF) queue sorting policy. For both scheduler configurations, we further use a best-fit task placement policy. We do not use a fixed resource environment to prevent bias when sampling or scaling traces [28]. Instead, we tailor the amount of available resources for each trace to reach roughly a 70 percent resource utilization on average, based on the amount of CPU (core) seconds of trace and its length. Although ambitious, 70 percent resource utilization is achievable in parallel HPC environments [49] and can be seen as a target for cloud environments. To evaluate the performance of each scheduler, we use three metrics commonly used to assess schedulers' performance [50], [51]: task response time (ReT), bounded task slowdown (BSD, using a lower bound of 1 second), and normalized workflow schedule length (NSL, the ratio between a workflow's response time and its critical path). The entire experiment, including software and data, can be reproduced on Code Ocean.[7]

We report the performance of each simulated scheduler in Table 6 per source. From this table we observe significant differences between schedulers and trace sources. In particular,

6. Available at https://github.com/atlarge-research/wta-sim
7. https://doi.org/10.24433/CO.8484557.v1

TABLE 6
The Performance in Simulation of Two Schedulers for Traces From Different Sources

Metric	Policy	Source of Trace: Askalon Old	Askalon New	Pegasus	Shell	SPEC
Avg. ReT	FCFS	$2.02 \cdot 10^5$ s	167 s	$2.43 \cdot 10^4$ s	9.76 s	491 s
	SJF	$1.74 \cdot 10^5$ s	113 s	$2.12 \cdot 10^4$ s	9.52 s	248 s
Avg. BSD	FCFS	$1.53 \cdot 10^4$	65.1	$1.31 \cdot 10^3$	1.13	47.4
	SJF	$0.14 \cdot 10^4$	11.6	$0.10 \cdot 10^3$	1.06	2.2
Avg. NSL	FCFS	$1.05 \cdot 10^5$	2.50	$2.35 \cdot 10^3$	1.12	13.9
	SJF	$0.01 \cdot 10^5$	3.14	$0.06 \cdot 10^3$	1.07	1.78

Lower values are better.

we find that the relative performance of schedulers differs between trace sources. For example, SJF outperforms FCFS on the normalized schedule length metric by up to two orders of magnitude on traces from Askalon Old and Pegasus. In contrast, on traces from Askalon New and Shell, the scheduling policies perform similarly. For other metrics, these differences are present, but less pronounced. SJF performs better than FCFS on response time and slowdown for each trace source, but the differences in performance between the schedulers vary greatly across traces.

Overall, we kept the working environment fixed per trace, yet obtained significantly different results depending on the scheduler and input trace. Thus, our trace-based simulations give practical evidence that researchers require experimenting with different traces to claim generality and feasibility of their proposed approaches.

C-2. Limited venue selection in the survey. Besides omitting venues that yielded no results on our initial query, we made sure that journals, workshops, and conferences were covered at various levels in term of quality. The selected venues are highly ranked in several of the available rankings, including CORE,[8] Google Scholar.[9] and AMiner.[10] As these rankings use different metrics to define the top-ranking, we made a selection that covers different types of venues that also match our experience in terms of quality, see the list in Table 1. We believe this covers the field of systems community to a degree where conclusions can be drawn from. We specifically focus on articles published in the systems communities as specialized communities, e.g., bioinformatics, focus on systems that solve domain-specific problems, but rarely conduct in-depth experiments, including trace-based, to test the system-level capabilities and behavior.

C-3. Level of data anonymization. The Google team published interesting work data [42], but their anonymization approach, of normalizing values of both resource consumption and available resources, reduces significantly the usability of traces and the characterization details they provide. We argue this type of anonymization is not preferred. When available resources per machine, e.g., available disk space, memory, etc., and resource consumption numbers are normalized, reusing traces for different environments becomes difficult. Researchers then need to make assumption on what kind of hardware the workflows were executed as done in the work of Amvrosiadis *et al.* [15] or need to assume a homogeneous environment. Instead, obfuscation techniques, such as multiplying both consumption and resources by a certain factor, allow for relative comparisons and the possibility to replay scheduling the workload on the resources while still concealing the original data.

C-4. The Workflow Trace Format. A fourth challenge is the properties included in the workload trace format. For each encountered property in other formats, we carefully decided whether to include it or not. Low-level details such as page caches are omitted to not complicate unnecessarily the traces. If future work demands change, the versioning schema per object will allow for these additions. In defining the fields of our trace format, we also looked at a variety of workflow specification languages and formalisms, from the very generic (e.g., BPMN/BPEL and Petri net) to the executable workflow formalisms (e.g., CWL and DAX).

6 Related Work

We survey in this section the relevant body of work focusing on trace archives and on characterizing workloads. Differently from other archives, the WTA focuses on *workloads of workflows*, preserving workflow-level arrival patterns and task inter-dependencies not found in other archives. Differently from other characterization work, ours is the first to reveal and compare workflow characteristics across different domains and fields of application.

Open-Access Trace Archives. Closest to this work is WorkflowHub [14], which archives traces of workflows executed with the Pegasus workflow engine and offers them in a unifying format containing structural information. WorkflowHub also provides a tool to convert Pegasus execution logs to traces, similar to our parsing tools. Different from this work, WorkflowHub's traces include a single workflow and thus not a workload with a job-arrival pattern. WorkFlowHub also does not provide statistical insights per trace and thus, they do not meet requirements **R-1** and **R-3**, and only partially meet **R-4**.

Also relatively close to this work, the ATLAS repository maintained by the Carnegie Mellon University [15] contains two traces (the S3 traces in this work), with other two traces announced but not yet released (as announced, the S7 traces in this work). None of their published traces contains task-interdependency data, so, although overlapping with our S3 and S7, the ATLAS work is different in scope and in particular does not address workflows. Further, they do not consider different domains nor fields, and their archive lacks a unified format, statistical insights, selection mechanisms, and tooling—thus, they do not meet our requirements **R1–4**.

Other trace-archives with similarities to this work include the MyExperiment archive (ME) [24], the Parallel Workloads

Archive (PWA) [52], and the Grid Workloads Archive (GWA) [53]. ME stores workflow executables, and semantic and provenance-data, but not provide execution traces as WTA does and thus has different scope. The PWA includes

8. http://portal.core.edu.au/conf-ranks
9. https://scholar.google.com/citations?view_op=top_venues&hl=en&vq=eng_computingsystems
10. https://cn.aminer.org/ranks/conf

traces collected from parallel production environments, which are largely dominated by tightly-coupled parallel jobs and, more recently, by bag-of-tasks applications. The GWA includes traces collected from grid environments; differently from this work, these traces are dominated by bag-of-tasks applications and by virtual-machine lease-release data.

Workload Characterization, Definition, and Modeling. There is much related and relevant work in this area, from which we compare only with the closely related; other characterization work does not focus on comparing traces by domain and does not cover a set of characteristics as diverse as this work, leading to so many findings. Closest to this work, the Google cluster-traces have been analyzed from various points of view, e.g., [54], [55], [56]. Amvrosiadis *et al.* [15], [23] compare the Google cluster traces with three other cluster traces, of 0.3-3 times the size and 3-60 times the duration, and find key differences; our work adds new views and quantitative data on diversity, through both survey and characterization techniques. Bharathi *et al.* [46] provide a characterization on workflow structures and the effect of workflow input sizes on said structures. Five scientific workflows are used to explain in detail the compositions of their data and computational dependencies. Using the characterization, a workflow generator for parameterized workflows is developed. Juve *et al.* [36] provide a characterization of six scientific workflows using workflow profiling tools that investigate resource consumption and computational characteristics of tasks. The teams of Feitelson and Iosup have provided many characterization and modeling studies for parallel [57], grid [58], and hosted-business [59] workloads; and Feitelson has written a seminal book on workload modeling [60]. In contrast, this work addresses in-depth the topic of workloads of workflows.

7 Conclusion and Ongoing Work

Responding to the stringent need for diverse workflow traces, in this work we propose the Workflow Trace Archive (WTA), which is an open-access archive containing workflow traces.

We conduct a survey of how the systems community uses workflow traces, by systematically inspecting articles accepted in the last decade in peer-reviewed conferences and journals. We find that, from all articles that use traces, less than 40 percent use realistic traces, and less than 15 percent use any open-access trace. Additionally, the community focuses primarily on scientific workloads, possibly due to the scarcity of traces from other domains. These findings suggest existing limits to the relevance and reproducibility of workflow-based studies and designs.

We design and implement the WTA around five key requirements. At the core of the WTA is an unified trace format that, uniquely, supports both workflow- and task-level NFRs. The archive contains a large and diverse set of traces, collected from 10 sources and encompassing over 48 million workflows and 2 billion CPU core hours.

Finally, we provide deep insight into the WTA traces, through a statistical characterization revealing that: (1) there are large differences in workflow structures between scientific, industrial, and engineering workflows, (2) our two biggest traces– from Alibaba and Google—have the most stable arrival patterns in terms of tasks per hour, (3) industrial workflows tend to have the highest level of parallelism, (4) the level of parallelism per domain is clearly divided, (5) engineering workloads tend to have the most tasks on the critical path, (6) the three domains inspected in this work show distinct critical path curves, (7) in order to claim generality of an approach, one should test a system with a variety of traces with different properties, possibly from different domains.

In ongoing work, we aim to attract more organizations to contribute real-world traces to the WTA, and to encourage the use of the WTA content and tools in educational and production settings. One of our goals is to develop a library system administrators can integrate into their systems to generate traces in our format. Our preliminary experience with this learns that developing such a library, even for a single system, requires significant engineering effort and is thus left for future work. We aim to support other formalisms in the future, including directed graphs, BPMN workflows, etc. based on the community's needs. Investigating if formalisms such as CWLProv can be used to further enhance the archive's content, possibly by merging, is another interesting item for future work. Finally, we aim to improve the trace format and statistics we report for each trace, based on community feedback.

Reproducibility Statement

We support reproducible science. A full description on how to reproduce our findings can be found in our technical report [38]. The WTA datasets are available online on the archive's website https://wta.atlarge-research.com/. The WTA tools, simulator, and parse scripts and survey data are available as free open-source software at https://github.com/atlarge-research/wta-tools, https://github.com/atlarge-research/wta-sim, and https://github.com/atlarge-research/wta-analysis, respectively. The experiment conducted in Section 5, **C-1** can be reproduced using our Code Ocean capsule available at https://doi.org/10.24433/CO.8484557.v1.

Acknowledgments

This work was supported by the projects Vidi MagnaData, Commit, the European Union's Horizon 2020 Research and Innovation Programme, under Grant 801091 "ASPIDE", and the National Science Foundation award 1664162.

References

[1] A. Ilyushkin *et al.*, "An experimental performance evaluation of autoscaling policies for complex workflows," in *Proc. 8th ACM/SPEC Int. Conf. Perform. Eng.*, 2017, pp. 75–86.
[2] F. Wu *et al.*, "Workflow scheduling in cloud: A survey," *J. Supercomputing*, vol. 71, pp. 3373–3418, 2015.
[3] P. K. Isom *et al.*, *Is Your Company Ready for Cloud*. Indianapolis, IN, USA: IBM Press, 2012.
[4] E. Deelman *et al.*, "The future of scientific workflows," *Int. J. High Perform. Comput. Appl.*, vol. 32, pp. 159–175, 2018.
[5] K. Weins, "Cloud computing trends: 2018 state of the cloud survey. rightscale," 2018.
[6] M. Kelly, "86 percent of companies use multiple cloud services," 2012. [Online]. Available: https://venturebeat.com/2012/05/10/cloud-services-data/

[7] A. Iosup *et al.*, "Massivizing computer systems: A vision to understand, design, and engineer computer ecosystems through and beyond modern distributed systems," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst.*, 2018, pp. 1224–1237.

[8] E. G. Coffman and R. L. Graham, "Optimal scheduling for two-processor systems," *Acta Inf.*, vol. 1, pp. 200–213, 1972.

[9] A. Slominski, "Adapting BPEL to scientific workflows," in *Proc. Workflows e-Sci.*, 2007, pp. 208–226.

[10] T. Murata, "Petri nets: Properties, analysis and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[11] Y. Gil *et al.*, "Examining the challenges of scientific workflows," *Computer*, vol. 40, no. 12, pp. 24–32, Dec. 2007.

[12] J. T. Dudley *et al.*, "In silico research in the era of cloud computing," *Nat. Biotechnol.*, vol. 28, no. 11, pp. 1181–1185, 2010.

[13] S. Cohen-Boulakia *et al.*, "Search, adapt, and reuse: the future of scientific workflows," *ACM SIGMOD Record*, vol. 40, pp. 6–16, 2011.

[14] R. F. da Silva *et al.*, "Community resources for enabling research in distributed scientific workflows," in *Proc. IEEE 10th Int. Conf. e-Sci.*, 2014, pp. 177–184.

[15] G. Amvrosiadis *et al.*, "On the diversity of cluster workloads and its impact on research results," in *Proc. USENIX Conf. Usenix Annu. Tech. Conf.*, 2018, pp. 533–546.

[16] L. Ramakrishnan *et al.*, "A multi-dimensional classification model for scientific workflow characteristics," in *Proc. 1st Int. Workflow Approaches New Data-Centric Sci.*, 2010, pp. 1–12.

[17] J. Cardoso *et al.*, "Workflow quality of service," in *Proc. Int. Conf. Enterprise Integration Modeling Technol.*, 2002, pp. 303–311.

[18] ACM, "Artifact review and badging," 2017. [Online]. Available: https://www.acm.org/publications/policies/artifact-review-badging

[19] B. Kitchenham *et al.*, "Systematic literature reviews in software engineering–a systematic literature review," *Inf. Softw. Technol.*, vol. 51, pp. 7–15, 2009.

[20] M. Ley, "The DBLP computer science bibliography: Evolution, research issues, perspectives," in *Proc. Int. Symp. String Process. Inf. Retrieval*, 2002, pp. 1–10.

[21] W. Ammar *et al.*, "Construction of the literature graph in semantic scholar," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technol.*, 2018, pp. 84–91.

[22] W. Schwiegelshohn, "How to design a job scheduling algorithm," in *Proc. Workshop Job Scheduling Strategies Parallel Process.*, 2014, pp. 147–167.

[23] G. Amvrosiadis *et al.*, "Bigger, longer, fewer: What do cluster jobs look like outside google?" Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-PDL-17–104, 2017.

[24] C. Goble *et al.*, "myExperiment: Social networking for workflow-using e-scientists," in *Proc. 2nd Workshop Workflows Support Large-Scale Sci.*, 2007, pp. 1–2.

[25] A. Iosup *et al.*, "The atLarge vision on the design of distributed systems and ecosystems," in *Proc. Int. Conf. Distrib. Comput. Syst.*, 2019. [Online] Available as *arXiv e-print*, https://arxiv.org/pdf/1906.07471.pdf

[26] A. Iosup *et al.*, "The openDC vision: Towards collaborative datacenter simulation and exploration for everybody," in *Proc. 16th Int. Symp. Parallel Distrib. Comput.*, 2017, pp. 85–94.

[27] D. Klusáček *et al.*, "On interactions among scheduling policies: Finding efficient queue setup using high-resolution simulations," in *Proc. Eur. Conf. Parallel Process.*, 2014, pp. 138–149.

[28] E. Frachtenberg and D. G. Feitelson, "Pitfalls in parallel job scheduling evaluation," in *Proc. Workshop Job Scheduling Strategies Parallel Process.*, 2005, pp. 257–282.

[29] L. Versluis *et al.*, "An analysis of workflow formalisms for workflows with complex non-functional requirements," in *Proc. ACM/SPEC Int. Conf. Perform. Eng.*, 2018, pp. 107–112.

[30] Amstutz *et al.*, "Common workflow language, v1. 0," Figshare, 2016.

[31] P. Couvares *et al.*, "Workflow management in condor," in *Proc. Workflows e-Science*, 2007, pp. 357–375.

[32] W. V. der Aalst *et al.*, "Advanced topics in workflow management: Issues, requirements, and solutions," *J. Integr. Des. Process Sci.*, vol. 7, 2003.

[33] F. Z. Khan *et al.*, "Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv," *GigaScience*, vol. 8, pp. 1–27, 2019.

[34] P. Xiong *et al.*, "vPerfGuard: An automated model-driven framework for application performance diagnosis in consolidated cloud environments," in *Proc. 4th ACM/SPEC Int. Conf. Perform. Eng.*, 2013, pp. 271–282.

[35] A. C. Zhou *et al.*, "A declarative optimization engine for resource provisioning of scientific workflows in IaaS clouds," in *Proc. 24th Int. Symp. High-Perform. Parallel Distrib. Comput.*, 2015, pp. 223–234.

[36] G. Juve *et al.*, "Characterizing and profiling scientific workflows," *Future Gener. Comput. Syst.*, vol. 29, pp. 682–692, 2013.

[37] Q. Sun *et al.*, "Adaptive data placement for staging-based coupled scientific workflows," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, 2015, Art. no. 65.

[38] L. Versluis *et al.*, "The workflow trace archive: Open-access data from public and private computing infrastructures – technical report," 2019, *arXiv:1906.07471*[cs.DC]. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2019arXiv190607471V

[39] Y. L. Simmhan *et al.*, "A framework for collecting provenance in data-centric scientific workflows," in *Proc. IEEE Int. Conf. Web Services*, 2006, pp. 427–436.

[40] W. Chen *et al.*, "Using imbalance metrics to optimize task clustering in scientific workflow executions," *Future Gener. Comput. Syst.*, vol. 46, pp. 69–84, 2015.

[41] S. Talluri *et al.*, "Characterization of a big data storage workload in the cloud," in *Proc. ACM/SPEC Int. Conf. Perform. Eng.*, 2019, pp. 33–44.

[42] C. Reiss *et al.*, "Obfuscatory obscanturism: making workload traces of commercially-sensitive systems safe to release," in *Proc. IEEE Netw. Operations Manage. Symp.*, 2012, pp. 1279–1286.

[43] D. S. Sayogo and T. A. Pardo, "Exploring the determinants of scientific data sharing: Understanding the motivation to publish research data," *Government Inf. Quart.*, vol. 30, pp. S19–S31, 2013.

[44] H. Bal *et al.*, "A medium-scale distributed system for computer science research: Infrastructure for the long term," *Computer*, vol. 49, no. 5, pp. 54–63, May 2016.

[45] D. Król *et al.*, "Workflow performance profiles: Development and analysis," in *Proc. Eur. Conf. Parallel Process.*, 2016, pp. 108–120.

[46] S. Bharathi *et al.*, "Characterization of scientific workflows," *Proc. 3rd Workshop Workflows Support Large-Scale Sci.*, 2008, pp. 1–10.

[47] A. Ilyushkin *et al.*, "Scheduling workloads of workflows with unknown task runtimes," in *Proc. 15th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput.*, 2015, pp. 606–616.

[48] A. Iosup *et al.*, "DGSim: Comparing grid resource management architectures through trace-based simulation," in *Proc. Eur. Conf. Parallel Process.*, 2008, pp. 13–25.

[49] J. P. Jones and B. Nitzberg, "Scheduling for parallel supercomputing: A historical perspective of achievable utilization," in *Proc. Workshop Job Scheduling Strategies Parallel Process.*, 1999, pp. 1–16.

[50] Y. K. Kwok and I. Ahmad, "Benchmarking and comparison of the task graph scheduling algorithms," in *Proc. 1st Merged Int. Parallel Process. Symp. Parallel and Distrib. Process.*, vol. 59, pp. 531–537, 1999.

[51] D. G. Feitelson *et al.*, "Theory and practice in parallel job scheduling," in *Proc. Workshop Job Scheduling Strategies Parallel Process.*, 1997, pp. 1–34.

[52] D. G. Feitelson, "Parallel workload archive," 2007. [Online]. Available: http://www. cs. huji. ac. il/labs/parallel/workload

[53] A. Iosup *et al.*, "The grid workloads archive," *Future Gener. Comput. Syst.*, vol. 24, pp. 381–422, 2008.

[54] C. Reiss *et al.*, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proc. 3rd ACM Symp. Cloud Comput.*, 2012, pp. 1–13.

[55] Y. Chen *et al.*, "Analysis and lessons from a publicly available google cluster trace," Univ. California, Berkeley, CA, Tech. Rep. UCB/EECS-2010–95, 2010.

[56] A.K. Mishra *et al.*, "Towards characterizing cloud backend workloads: insights from google compute clusters," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 37, pp. 34–41, 2010.

[57] D. G. Feitelson *et al.*, "Experience with using the parallel workloads archive," *J. Parallel Distrib. Comput.*, vol. 74, pp. 2967–2982, 2014.

[58] A. Iosup and D. Epema, "Grid computing workloads," *IEEE Internet Comput.*, vol. 15, no. 2, pp. 19–26, Mar./Apr. 2011.

[59] S. Shen *et al.*, "Statistical characterization of business-critical workloads hosted in cloud datacenters," in *Proc. 15th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput.*, 2015, pp. 465–474.

[60] D. G. Feitelson, *Workload Modeling for Computer Systems Performance Evaluation*. Cambridge, U.K.: Cambridge Univ. Press, 2015.

Laurens Versluis received the BSc and MSc degrees in computer science from the Delft University of technology, The Netherlands. Currently, he is working toward the PhD degree in the Massiving Computer Systems Group, Department of Computer Science, Faculty of Sciences, VU Amsterdam, The Netherlands. His research interests include cloud computing, distributed systems, scheduling, complex workflows, image processing, and privacy enhancing technologies.

Roland Mathá received the BSc degree in computer science and the MSc degree from the University of Innsbruck, Austria, in 2011 and 2014, respectively. Since January 2015, he is working toward the PhD degree under the guidance of prof. Radu Prodan. His research interests include Cloud simulations, workflow applications, and multi-objective optimisations.

Sacheendra Talluri received the MSc degree from the Delft University of Technology, The Netherlands. In the spring of 2018, he was a research intern at big data company Databricks, working on resource management and scheduling across the memory-storage stack.

Tim Hegeman received the BSc and MSc degrees in computer science from the Delft University of Technology, The Netherlands. He is currently working toward the PhD degree at Vrije Universiteit Amsterdam, The Netherlands under guidance of prof. Alexandru Iosup. His research interests include distributed systems, big data, and performance engineering.

Radu Prodan received the PhD degree from the Vienna University of Technology, in 2004. He was a associate professor until 2018 at the University of Innsbruck, Austria. He is currently a professor in distributed systems at the Institute of Software Technology, University of Klagenfurt. He has participated in numerous national and European projects, as is currently principal coordinator of the H2020-ICT project ARTICONF (smART socIal media eCOsystem in a blockchaiN Federated environment). He is the author of one book, more than 100 journal and conference publications, and is the recipient of two IEEE best paper awards.

Ewa Deelman received the PhD degree in computer science from Rensselaer Polytechnic Institute. She is currently a research professor with the Computer Science Department and the assistant director for the Science Automation Technologies group at the University of Southern California Information Sciences Institute. Her research focuses on distributed computing, in particular regarding how to best support complex scientific applications on a variety of computational environments, including campus clusters, grids, and clouds.

Alexandru Iosup received the PhD degree in computer science from TU Delft, The Netherlands, in 2009. He is currently a tenured full professor and University Research chair with the Vrije Universiteit Amsterdam, The Netherlands. He is also chair of the SPEC Research Cloud Group. He was awarded the yearly Netherlands Prize for Research in Computer Science (2016), the yearly Netherlands Teacher of the Year (2015), and several SPECtacular awards (2012–2017). His research interests include massivizing computer systems, that is, making computer systems combine desirable properties such as elasticity, performance, and availability, yet maintain their ability to operate efficiently in controlled ecosystems. Topics include cloud computing and big data, with applications in big science, big business, online gaming, and (upcoming) massivized education.